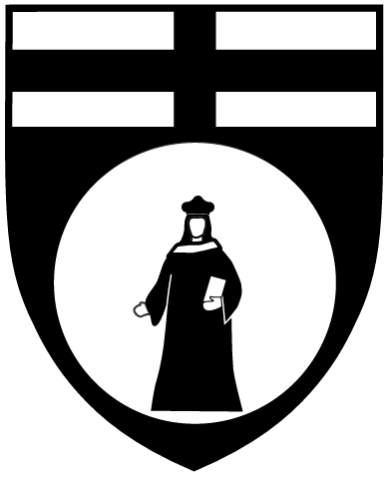


Dibris Dipartimento di Informatica, Bioingegneria,
Robotica e Ingegneria dei Sistemi

Virtual Reality For Robotics

Project Report

Photogrammetry & 3D Model Reconstruction

Advisor: Prof. Gianni Viardo Vercelli

UNIVERSITY OF GENOA, JUNE 2023

Member list & Workload

No.	Full name	Student ID	Percentage of work
1	Ammar Iqbal	5183355	25%
2	Ankur Kohli	5160903	25%
3	Basit Akram	5161322	25%
4	Naveed Manzoor Afridi	5149575	25%

Contents

1 Introduction

In the past three decades, Virtual reality (VR) has been evolving at a rapid pace, allowing for the creation of worlds that replicate the actual world and the creation of interactions and visualizations that would otherwise be impossible.

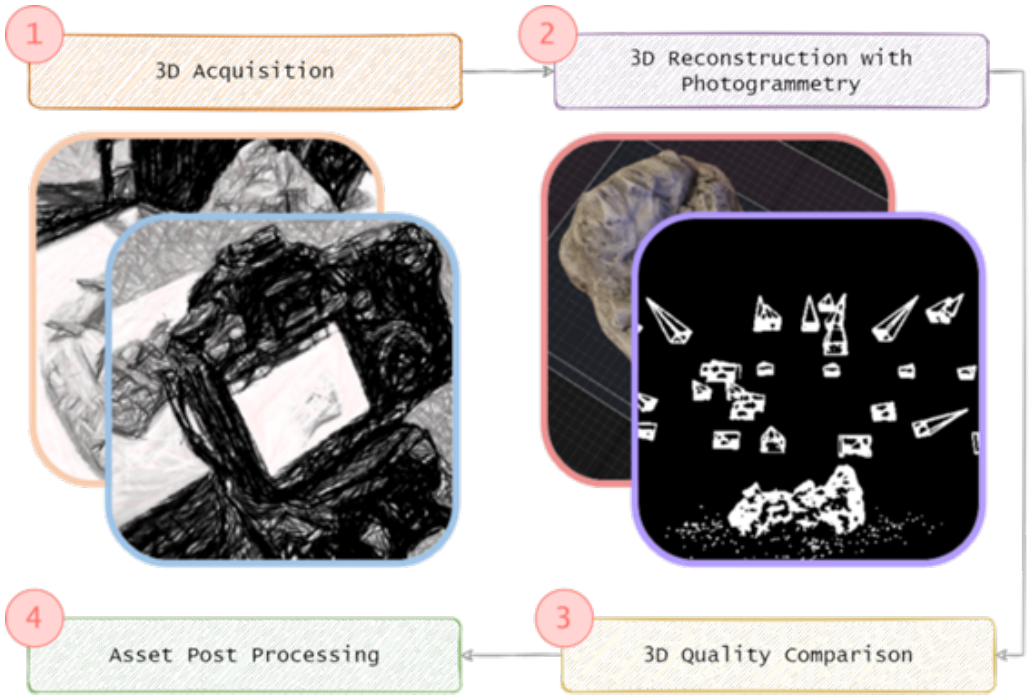
Virtual environments for cultural heritage diffusion and virtual experimentation were created by integrating 3D image acquisition techniques such as digital photogrammetry (captured reality) with computer 3D modelling.

Many sophisticated applications now demand three-dimensional 3D data. The challenge of efficiently and precisely constructing real-time 3D representations of a scene has applications in various computer vision domains, including robotics, augmented and virtual reality, 3D image processing, 3D visualization and entertainment. [7] At present, the computer vision scientific community has developed novel reconstruction methods that employ various types of equipment. In the examination of dynamic or static settings, the third dimension is critical. Surveillance and robotic domains that use depth information to acquire a much better understanding of the environment are examples of fields of use in everyday life that may leverage the third dimension.

Existing 3D modelling methods may be classified based on the needed input data and their efficacy is measured by the number of scenes that can be processed, the quality of the final model, and the overall processing time. Depending on the user's needs, automated, semi-automatic, or manual image-based methodologies can be used to generate digital models suitable for inspections, visualization, or documentation. Automated approaches focus primarily on process automation, although the outputs are often enough for nice-looking real-time 3D recording or basic visualization. Semi-automated solutions, on the other hand, attempt to strike a compromise between precision and automation and are extremely valuable for detailed documentation and restoration planning. [8]

Laser scanning technology has evolved as an effective and competitive option for making 3D reconstructions in recent years. The primary benefits of using this technology are (i) speed, (ii) accuracy, and (iii) reconstruction resolution. Furthermore, the scanners' field of vision enables the reconstruction of objects ranging in size from a few centimetres to several meters and existing across a short or long distance. As a result, this technique is appropriate for large-scale sceneries such as the interior and exterior of buildings, and it is widely regarded by the community as a viable support for the documenting and protection of historic structures, monuments, or archaeological sites. [9] The following phases are required for epigrammatically reconstructing a scenario:

1. Acquiring enough colour range scans to adequately cover the 3D scene
2. 3D Reconstruction with Photogrammetry
3. 3D Quality Comparison
4. Data processing for final 3D surface model refining



: 3D model Reconstruction Phases [5]

3D Data Acquisition: 3D data acquisition refers to the process of capturing information about the three-dimensional shape, surface, and/or appearance of an object or scene using specialized equipment or techniques. This can be done using a variety of methods, such as structured light, laser scanning, photogrammetry, and computer vision, among others. The captured data can then be used for a variety of purposes, such as computer graphics, scientific visualization, measurement, and inspection.

3D Data Processing: 3D data processing refers to the process of using computer software to analyze and manipulate 3D data captured by a reality capture device, such as a laser scanner or photogrammetry camera. This can include tasks such as cleaning and filtering the data, aligning and merging multiple scans or images, and creating a 3D model or point cloud of the captured environment. The resulting 3D data can be used for a variety of purposes, including building and construction, asset management, heritage preservation, and more.

2 State of Art

Photogrammetry is a technique for creating three-dimensional models and maps from two-dimensional images. It involves taking a series of photographs from different angles and using specialized software such as Capture Reality [1] and Unreal Engine [4] to process the images and generate a 3D model or map. [5]

One of the key challenges in photogrammetry is accurately aligning the images, which is known as image registration. This can be difficult due to factors such as variations in lighting, camera position, and image distortion. To address these challenges, researchers have developed techniques such as feature-based alignment and structure-from-motion.

Another important aspect of photogrammetry is the creation of accurate and detailed 3D models. This can be achieved through the use of techniques such as multi-view stereo, which involves comparing multiple images of the same scene to generate a 3D point cloud. Other techniques, such as lidar and structured light, can also be used to create high-resolution 3D models. [6]

Recent advances in machine learning have also led to the development of neural network-based methods for photogrammetry. These methods can learn to align images and create 3D models from large datasets, and have the potential to significantly improve the accuracy and speed of photogrammetry.

Overall, photogrammetry and 3D model reconstruction are active areas of research with numerous practical applications, including mapping, architectural documentation, and heritage preservation.

2.1 Sample: Open Data and Linked Open Data

The state of the art of photogrammetry and 3D model reconstruction in virtual reality for robotics has been significantly advanced by leveraging open data and linked open data. Researchers can access a wide range of freely available data, such as satellite imagery and geospatial data, to create accurate 3D models of real-world environments. By interlinking data from various sources, additional contextual information can be incorporated into the virtual environment, enhancing realism and enabling informed decision-making. Collaboration and knowledge sharing are facilitated through the sharing of data sets and methodologies, promoting innovation in the field. Open data and linked open data also support the development of machine learning algorithms for more efficient and accurate reconstruction. Overall, these approaches drive advancements in photogrammetry, 3D model reconstruction, and virtual reality for robotics.

2.2 Sample: Cultural heritage ontologies

The state of the art of photogrammetry and 3D model reconstruction in virtual reality for robotics has been significantly influenced by the use of cultural heritage ontologies. Cultural heritage ontologies provide a structured framework for organizing and representing knowledge about cultural artifacts, historical sites, and heritage

objects. By applying ontologies to the field of photogrammetry and 3D model reconstruction, researchers can enhance the interpretation and understanding of cultural heritage in virtual reality environments. Ontologies enable the semantic annotation and classification of 3D models, allowing for meaningful associations and relationships between different cultural objects. This enriched information enhances the immersive experience and educational value of virtual reality applications in the cultural heritage domain. It also enables more sophisticated interactions with the virtual environment, such as virtual tours, interactive exhibitions, and historical reconstructions. Cultural heritage ontologies thus contribute to preserving and promoting cultural heritage through the seamless integration of photogrammetry, 3D model reconstruction, and virtual reality technologies.

3 Tools

Here is the list of tools that we encounter to fulfil the requirements of our project along with the GitHub repository.
1. Reality Capture [1]

Reality Capture is a leading software tool used in the field of photogrammetry and 3D model reconstruction. It offers advanced capabilities for capturing and processing images to create highly detailed and accurate 3D models of real-world objects and scenes. Here are some key features and functionalities of Reality Capture:

1. **Image-based Reconstruction:** Reality Capture uses a collection of images taken from different viewpoints to reconstruct 3D models. It employs advanced algorithms to align and stitch these images together, extracting geometric and textural information to create a 3D representation of the scene.
2. **Multiple Input Sources:** The software supports various input sources, including standard RGB images, laser scans, and structured light scans. It can seamlessly integrate data from different sources to create a comprehensive and detailed 3D model.
3. **Automatic Alignment and Calibration:** Reality Capture automates the alignment and calibration process, making it easier to handle large datasets. The software identifies common features in the images and automatically aligns them to create an accurate representation of the scene.
4. **Dense Point Cloud Generation:** By leveraging multi-view stereo algorithms, Reality Capture generates dense point clouds, capturing intricate details and providing a solid foundation for the 3D model reconstruction process.
5. **High-Quality Texturing:** The software enables high-quality texturing of the reconstructed 3D models by applying the original images' textures onto the model's surface. This results in realistic and visually appealing representations of the captured scene.
6. **Mesh Generation and Optimization:** Reality Capture generates optimized meshes from the dense point clouds, producing a continuous and watertight 3D surface. The software employs various optimization techniques to refine the mesh and enhance its visual quality.
7. **Export and Integration:** The 3D models created in Reality Capture can be exported in various file formats, including industry-standard formats such as OBJ, FBX, and PLY. This allows for seamless integration with other software tools and platforms used in 3D visualization, virtual reality, and robotics.
8. **Scalability and Performance:** Reality Capture is designed to handle large-scale datasets efficiently. It leverages advanced processing techniques and can utilize multiple computing resources, such as GPUs, to accelerate the reconstruction process and achieve high-performance results.
9. **User-Friendly Interface:** The software provides an intuitive and user-friendly interface, making it accessible to both experts and beginners. It offers a range of automated tools and guided workflows that streamline the reconstruction process and make it easier to achieve high-quality results.

Overall, Reality Capture is a powerful and versatile tool for photogrammetry and 3D model reconstruction. Its advanced capabilities, automation features, and scalability make it a popular choice among professionals in industries such as architecture, archaeology, gaming, visual effects, and robotics.

2. Unreal Engine 5 [4]

Unreal Engine is a popular and powerful real-time 3D engine and development platform created by Epic Games. It provides a comprehensive suite of tools and features for creating interactive and immersive experiences across various platforms, including video games, virtual reality (VR), augmented reality (AR), architectural visualization, and more. Here are some key aspects and features of Unreal Engine:

1. **Real-time Rendering:** Unreal Engine offers advanced real-time rendering capabilities, allowing developers to create stunning and realistic visuals. It supports high-quality materials, lighting, shadows, and post-processing effects to achieve lifelike graphics.
2. **Blueprints Visual Scripting:** Unreal Engine includes a visual scripting system called Blueprints, which enables non-programmers to create gameplay mechanics, interactive elements, and complex behaviors using a node-based interface. It provides a user-friendly way to create game logic without writing code.
3. **Content Creation Tools:** The engine provides a range of powerful tools for content creation, including a robust editor for designing levels and environments, a material editor for creating and editing materials, a particle system for visual effects, and a physics engine for realistic simulations.
4. **Cross-platform Development:** Unreal Engine supports cross-platform development, allowing developers to create applications for PC, consoles, mobile devices, VR headsets, and AR platforms. It provides built-in support for major platforms, simplifying the process of deploying projects to different devices.
5. **Blueprint Visual Scripting:** Unreal Engine includes a visual scripting system called Blueprints, which enables non-programmers to create gameplay mechanics, interactive elements, and complex behaviors using a node-based interface. It provides a user-friendly way to create game logic without writing code.
6. **Marketplace and Community:** Unreal Engine has a thriving marketplace where developers can access a wide range of assets, including 3D models, materials, animations, and plugins, to enhance their projects. The engine also has a strong community of developers who actively share knowledge, provide support, and contribute to the growth of the engine.
7. **Performance and Optimization:** Unreal Engine offers various optimization features and techniques to ensure high-performance execution. This includes tools for profiling and debugging, GPU and CPU optimization, and efficient memory management.
8. **Blueprint Visual Scripting:** Unreal Engine includes a visual scripting system called Blueprints, which enables non-programmers to create gameplay mechanics, interactive elements, and complex behaviors using a node-based interface. It provides a user-friendly way to create game logic without writing code.

9. **Industry Adoption:** Unreal Engine is widely adopted in the gaming industry and beyond. It has been used to develop highly acclaimed games, VR experiences, animated films, architectural visualizations, and training simulations. Its versatility and feature set make it a popular choice for a wide range of projects.

Unreal Engine continues to evolve and introduce new features with regular updates, empowering developers to create cutting-edge and immersive experiences across multiple industries.

For this project we used **Unreal Engine version 4.25.4**.

3. GitHub Project

4 Description

The project of Photogrammetry and 3D Model Reconstruction in terms of Virtual Reality for Robotics aims to leverage the power of photogrammetry and 3D model reconstruction techniques in the context of virtual reality (VR) to enhance robotics applications. The project involves capturing images of the real-world environment using cameras or other imaging devices and using photogrammetry algorithms to reconstruct a detailed and accurate 3D model of the scene.

The 3D model, along with the captured images, is then integrated into a virtual reality environment, creating a realistic and immersive virtual representation of the physical space. This virtual environment serves as a simulation platform where robotic systems can be tested, trained, and interacted with.

The project focuses on developing advanced techniques to handle the challenges specific to robotics applications, such as real-time reconstruction, dynamic scene handling, and accurate object tracking. It explores the integration of multiple sensors, such as RGB cameras, depth sensors, and inertial measurement units (IMUs), to improve the accuracy and reliability of the 3D reconstructions.

In addition, the project aims to develop interactive user interfaces and control mechanisms within the virtual reality environment, enabling users to manipulate virtual objects, control robotic systems, and receive haptic feedback. The integration of robotics frameworks, such as robot simulators or control systems, with the virtual environment is also a key aspect of the project.

The ultimate goal of the project is to create a seamless integration between the physical and virtual worlds, allowing robotic systems to perceive and interact with the virtual environment in a realistic and intuitive manner. This integration opens up possibilities for various robotics applications, including robot training, task planning, autonomous navigation, and human-robot interaction.

Through this project, the potential of photogrammetry and 3D model reconstruction in the context of virtual reality is harnessed to advance the field of robotics and pave the way for more sophisticated and immersive robotic applications in diverse industries.

4.1 Reality Capture Steps

Firstly, we will go through the **data acquisition** set-up and the **data processing** steps, showing the workflow with a paying option (Reality Capture).

- **Step1: Data Acquisition:** The necessary equipment is easy to get by. Only a camera is needed (your smartphone works), and a tripod can be used if more stability and control are needed, but it is possible to work without it. The camera used is a Canon EOS 50D with a Canon Ultrasonic lens. The photos are

saved in JPG format with a size of 4752x3168 pixels. Before starting the acquisition, we take a few photos to determine what settings should be used. In our experiments, we set the camera parameters as follow:

- **Focal length:** A fixed focal length of 28 mm, giving a good balance between distance to the subject and workable area;
- **ISO:** 400, which is a good value to start with, but if your scene appears too dark, you can push this value up to 1600;
- **Diaphragm opening:** F/3.5, which allows getting enough light in at the cost of a limited depth of field. In our case, as we focus only on a tiny object, we are good here.
- **Shutter speed:** 125 ms (1/8 s), which our tripod allows thanks to more stability. Without tripod, below 1/200s is super tricky;
- **White balance:** 4000 K. This adapts to our scene's ambient lighting.

The above parameters remain the same throughout the acquisition process.

The acquisition strategy in itself is straightforward. We place the camera on the tripod and adjust the height and the angle so that the object completely fits in the field of view. In this case, as the object is relatively low, we can give a downward angle to the camera so that we can also shoot the top part. We use autofocus to ensure the photos won't be blurry, then swap to manual mode.

We then move the tripod along a circular trajectory all around the object. We need to remain at a constant distance from the object so that the focus stays right. We check the photos as we go. Once we have achieved the first circle around the object, we diminish the tripod's legs and lower the downward angle of the camera. We adjust the focus and then swap to manual mode again. We take a few more photos all around the subject.

Finally, we remove the camera from the tripod and take a vertical photo of the complete object. To do this, we adjust the focus again. Then, we take detailed photos to ensure that the texture will be fine on all parts of the object. The idea is basically to take enough photos to allow a complete and detailed reconstruction while keeping in mind that an unnecessarily large number of inputs will only slow down and complicate the process. ideally, for such an object, you have 25 pictures (8 horizontal positions multiplied by three layers of heights, plus one on top).

- **Step2: Photogrammetry Processing:**

  The photos are first sorted to eliminate images of bad quality (blurry, out of frame, wrong exposition) that would only disturb the next steps.

  Now that all the testbench is clear, let us dive into the 3D photogrammetry processing details through a workflow with Reality Capture.

The choice toward Reality Capture is pretty straightforward: we can do everything for free. As long as nothing is exported. So, nothing better than showing State-of-the-Art Professional Software. But for this project we used licensed of a Reality Capture which was provided by our Prof. Gianni Viardo Vercelli & Prof. Saverio Iacono. Below are are the steps to be followed while developing reconstruction of 3d model in Reality Capture:

- The first step is to create a new project and import our approx 1000 images. This is done by drag-and-drop or clicking on the "Inputs" icon as shown in figure below:

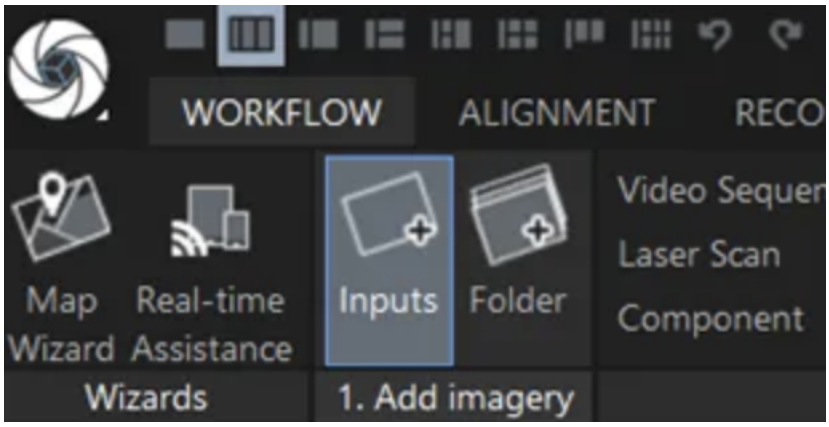


Figure 1: Data Import

- Then, we start the **alignment** with the first icon in the "Alignment" tab. For this project, we keep the default parameters. Below is the figure shown about alignment:

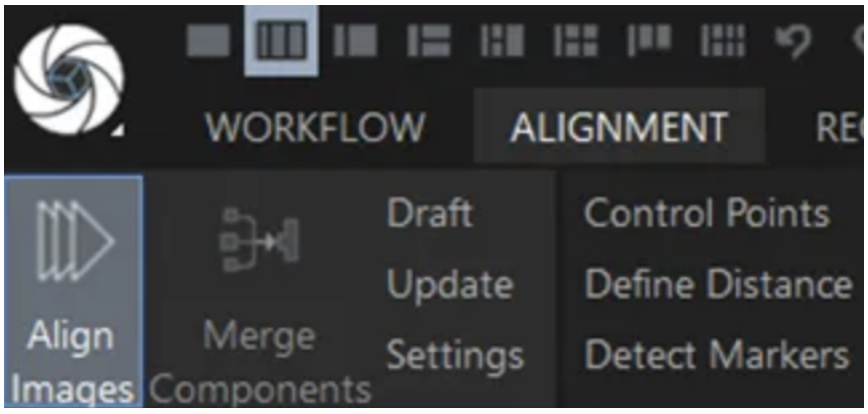


Figure 2: Alignment

- Furthermore, we imported our approx 1000 images and change the **alignment settings** before starting

the process. We set the “Number of features detected per image” and “Number of features detected per mpx” (megapixel) parameters to 100k and set the “Image overlap” to “High” .

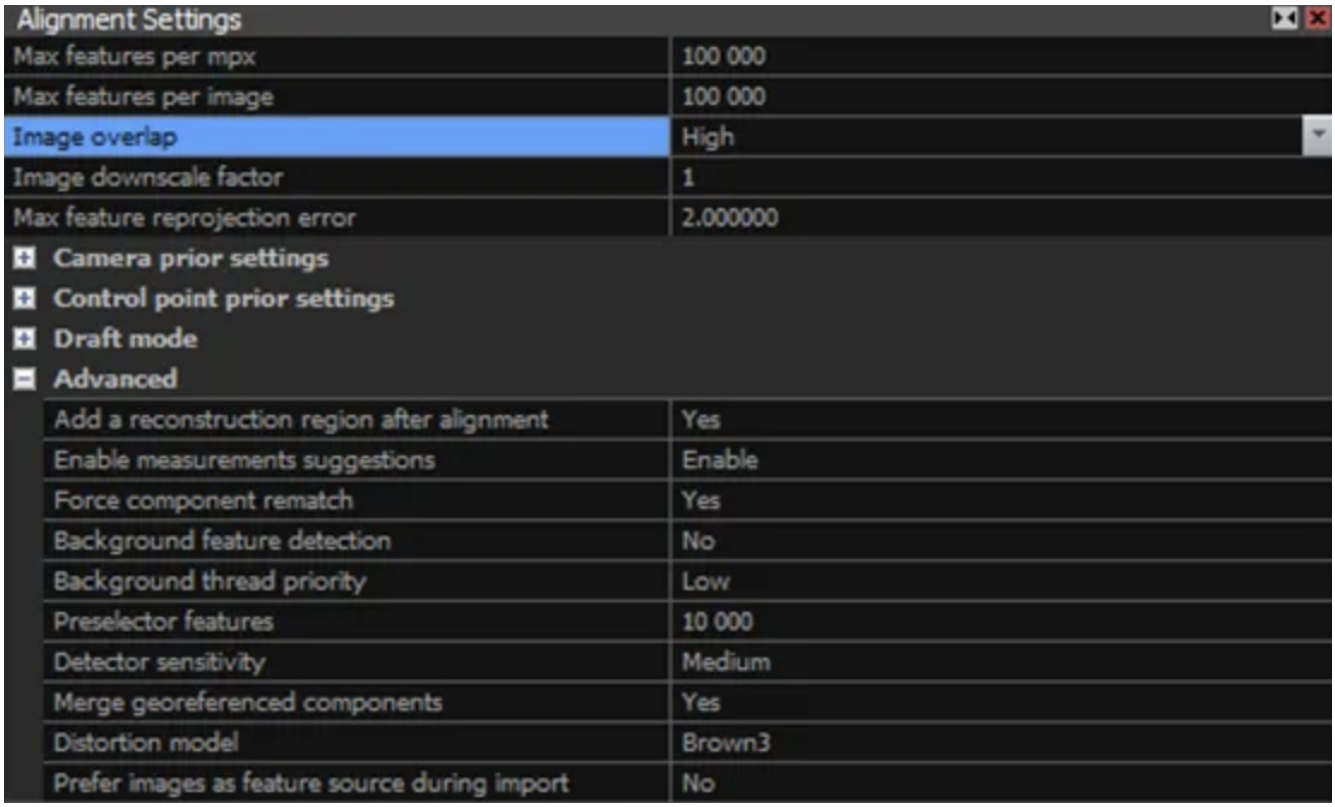


Figure 3: Alignment settings that work well in complex cases with good-quality datasets.

- Given the low number of inputs, the alignment is done in only 30 seconds. After this step, we have a **component** that consists of a point cloud and the estimated positions of the cameras.

  We can check that all inputs are aligned and see that the positions of 48 cameras out of 48 are estimated. We also check the **alignment report**. This report provides the alignment duration, the number of aligned inputs, the point count, and the mean and maximal re-projection error in pixels. It is also possible to retrieve the alignment settings that were used. The figure below depicts alignment report:

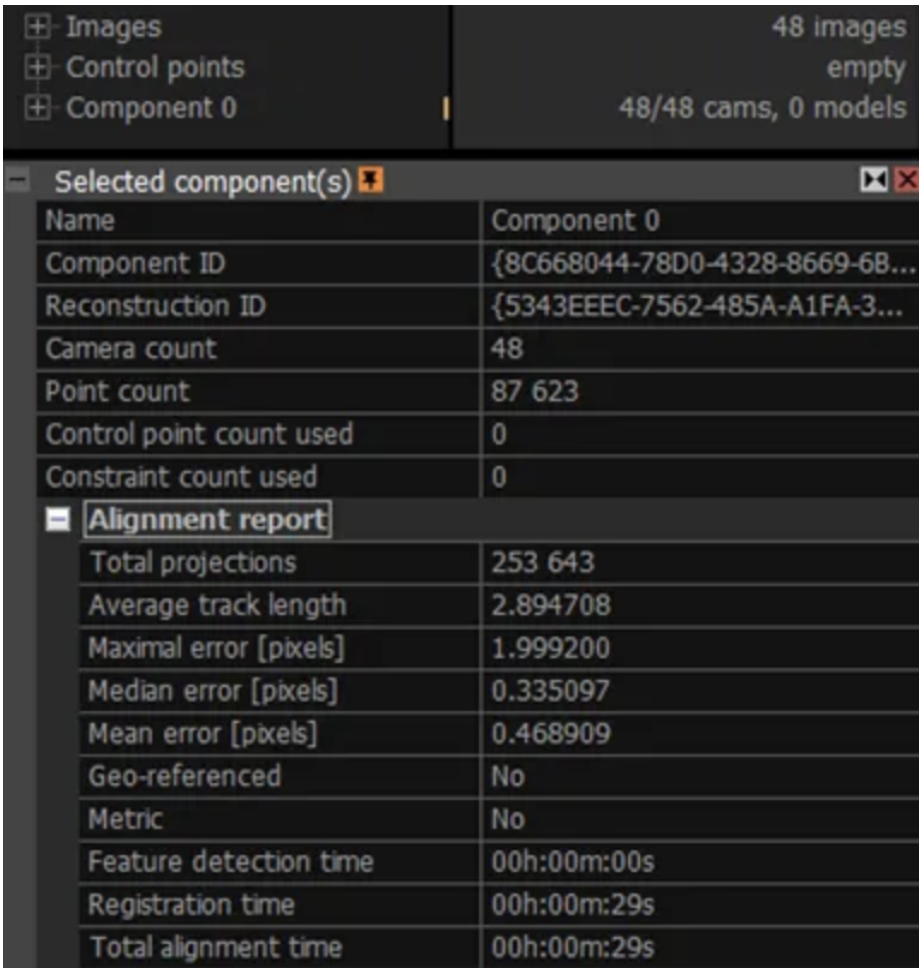


Figure 4: Alignment Report

- Also, we go with the **reconstruction settings** steps before building the model. We start a reconstruction in Normal detail with default parameters, except for the "Maximal vertex count by part," which is set on 3M. In "Normal detail," the value of the "Image downscale" is two by default.

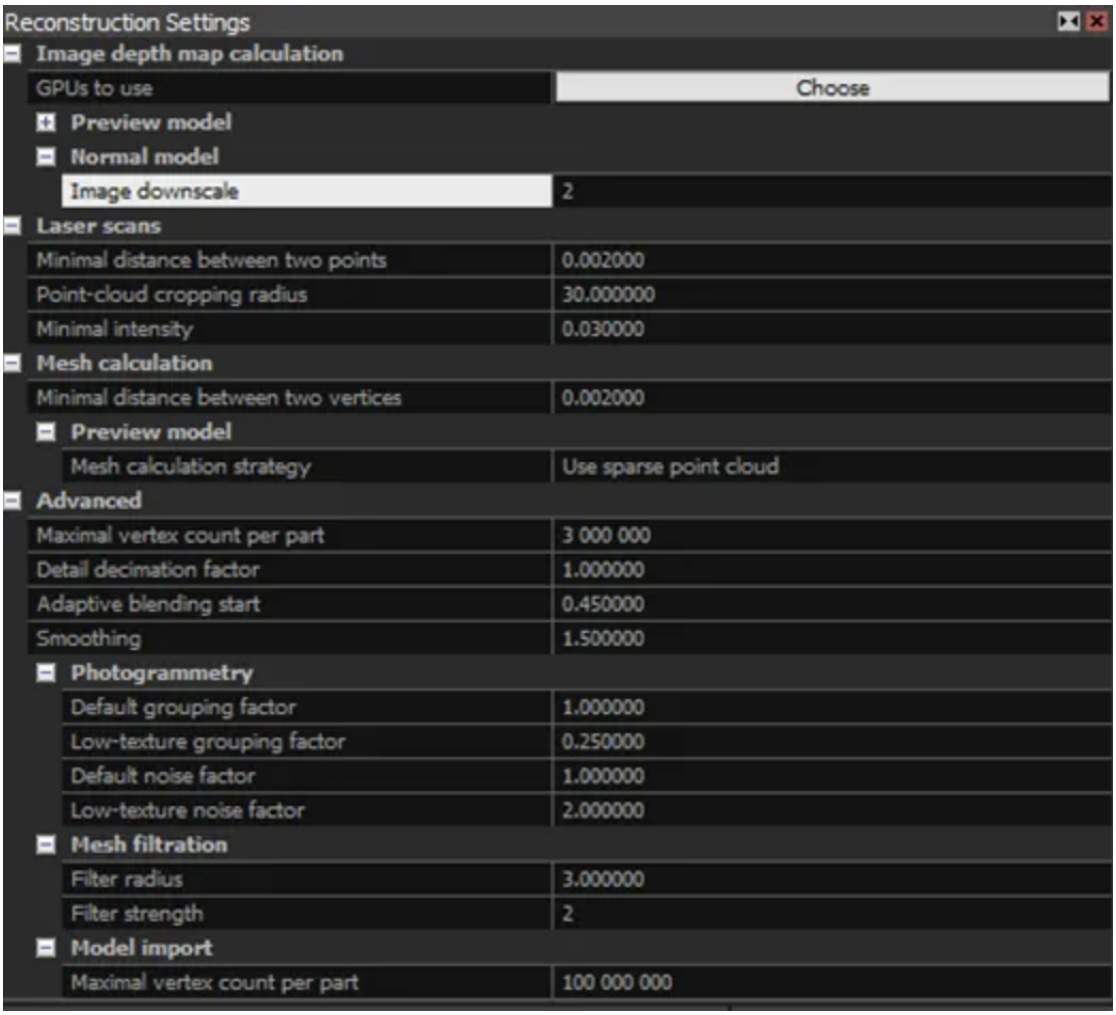


Figure 5: Reconstruction Settings

- Moreover, a visual check of the point cloud is done. If everything is fine, we can proceed to the next step, the **reconstruction**. We keep the default parameters, except for the **"Image downscale"** factor, which is set to 2.

  The **reconstruction** is started by clicking on the "Normal detail reconstruction" icon in the "Reconstruction" tab. This step takes only five minutes, resulting in a mesh of around 1 million triangular facets. It is a lot compared to the number of countries on earth (195 in 2023), but very tiny compared to what we usually work with. With the "Advanced" **selection** tool, we select the marginal and large triangles, then filter the selection. We also adjusted the reconstruction box to eliminate the parts that belong to the furniture on which the object was placed ("Box" selection tool). The figure below highlights reconstruction

of the model:

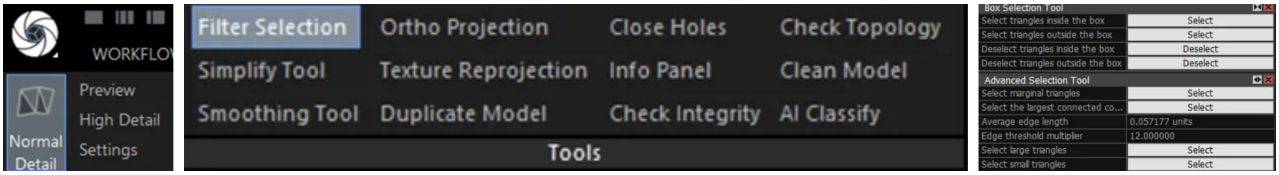


Figure 6: Reconstruction button (Left), Filter Selection tool (Middle) and the possible filtering options (right)

- Now, we proceed with the smoothing step. The **smoothing** is done on the surfaces and not the edges, 100 iterations of the algorithm are performed, and the weight is set to 0.5. Smoothing is finished in 10 min.

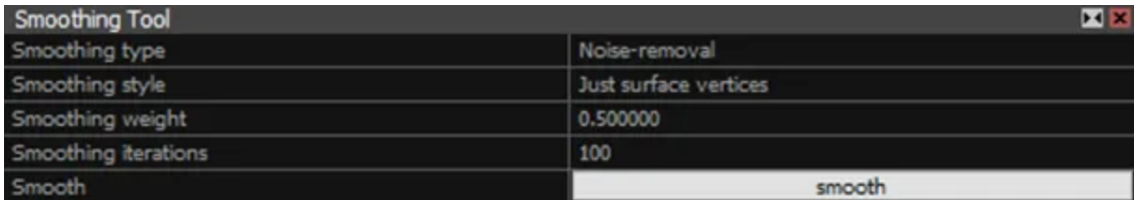


Figure 7: Smoothing Tool Settings

These parameters ensure efficient smoothing without losing much detail.

After aforementioned steps, we build a model and this model is about **151 millions triangles** and this is maximum as shown in figure figure 8.

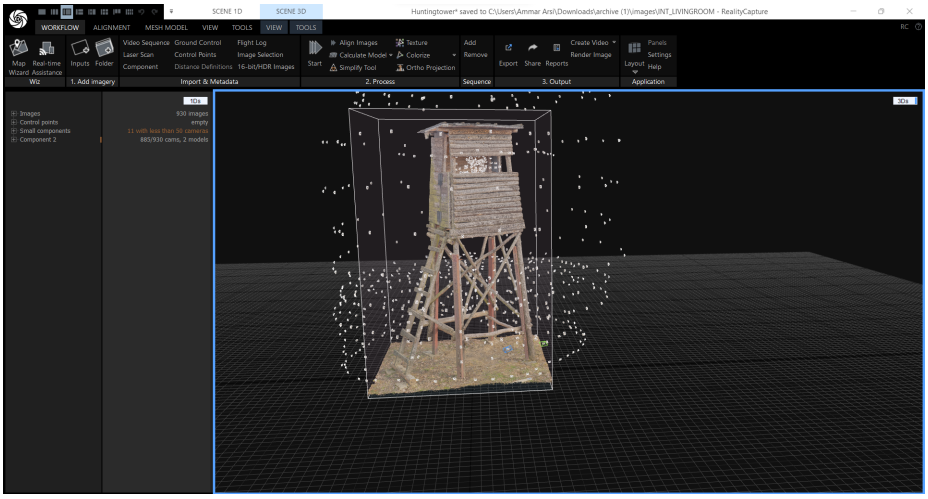


Figure 8: 151 millions triangles on our model

So, finally we simplified the model and build up to **5 millions triangles** as shown in figure 9,10, & 11.

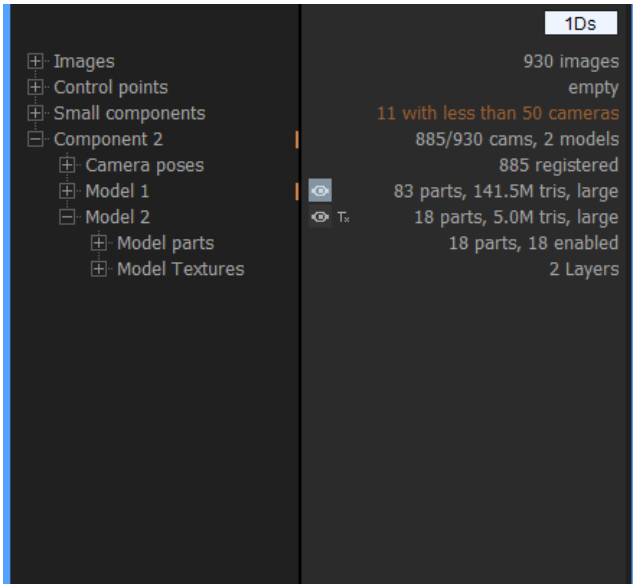


Figure 9: Scale Down Triangles: Step1

Figure above shows Scale Down Triangles: Step1 and figure below shows Scale Down Triangles: Step2.

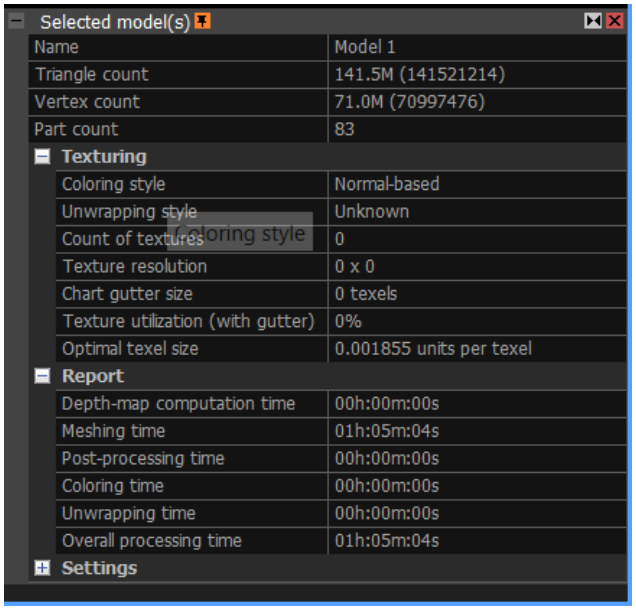


Figure 10: Scale Down Triangles: Step2

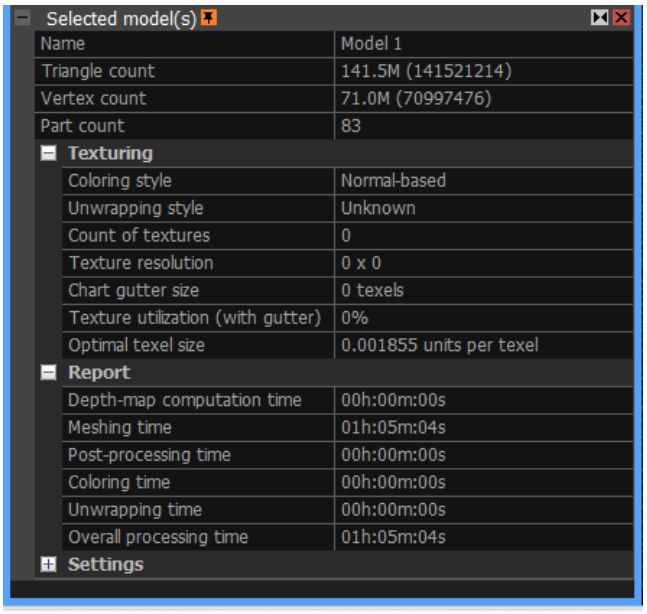


Figure 11: Scale Down Triangles: Step3

Figure above shows the final step for scale down triangles in a model.

- Another tool that may be used at this point is the **"Simplify Tool"**. It is possible to simplify the model to reach a fixed number of triangles (type: "Absolute") or to retain a certain percentage of triangles.

Figure 12: Simplify Tool Settings

- The last step is to **texture** the mesh. To do this, we keep the default parameters and click on the "Texture" button in the "Reconstruction" tab. By default, a downscale factor of 2 is used for texturing. The figure below explains Texture Menu:

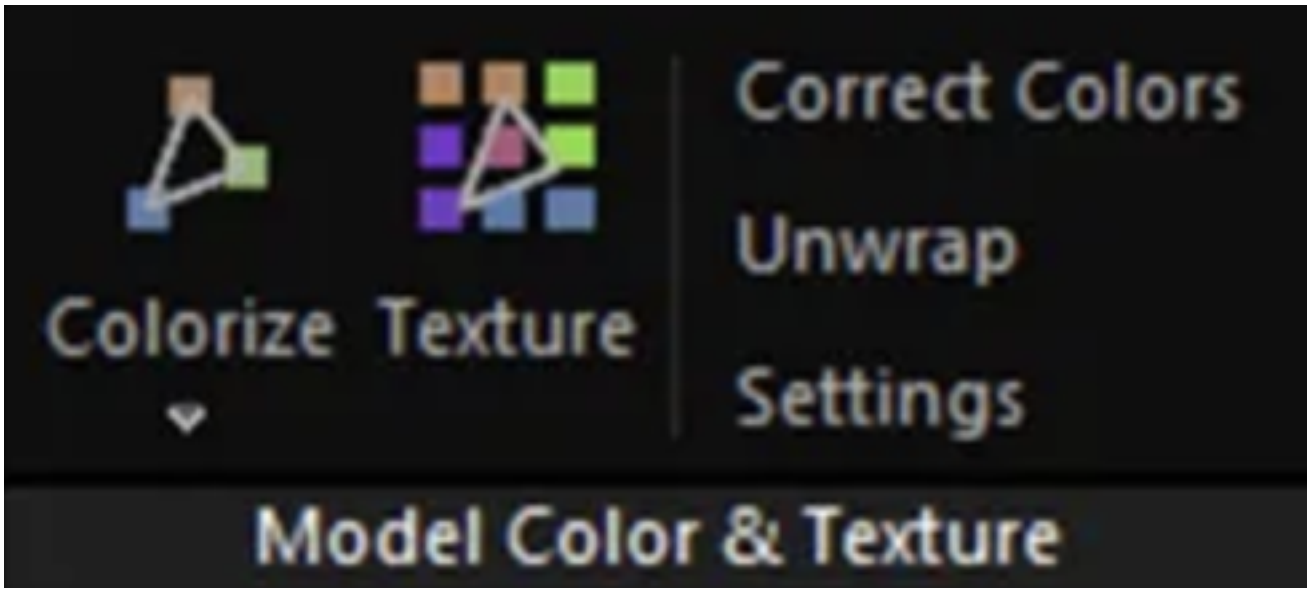


Figure 13: Texture Menu

After, texturing we imported our model into Unreal Engine with an extension of ***.fbx*** *which is also a Solid Works compatible file.*

4.2 Importing Model from Reality Capture to Unreal Engine

Importing models from Reality Capture to Unreal Engine involves several steps. Here's a general overview of the process:

- **Model Export from Reality Capture:**
  - In Reality Capture, ensure that your 3D model is complete and processed.
  - Export the model in a compatible file format. Unreal Engine supports various formats such as FBX, OBJ, and Alembic.
  - Specify the desired export settings, including scale, vertex color preservation, and material information.
- **Prepare Unreal Engine Project:**
  - Launch Unreal Engine and open your project.
  - Create a new folder within the Content Browser to store your imported models.
- **Import the Model into Unreal Engine:**
  - In the Content Browser, navigate to the folder where you want to import the model.

- Right-click and select "Import" to open the import dialog.
- Browse for the exported model file and select it.
- Configure the import settings based on your requirements, such as scale, materials, and LOD (Level of Detail) options.
- Click "Import" to begin the import process.

- **Configure Material and Texture:**
  - After the import is complete, locate the imported model in the Content Browser.
  - Double-click on the model to open the Static Mesh Editor.
  - In the Static Mesh Editor, you can assign materials and textures to the model.
  - Create or import the materials and textures you want to use, and then apply them to the corresponding surfaces of the model.
- **Place the Model in the Scene:**
  - Drag and drop the imported model from the Content Browser into the desired location in the scene or level.
  - Adjust its position, rotation, and scale as needed.
- **Adjust Collision and LODs:**
  - In the Static Mesh Editor, you can define the collision properties of the model.
  - Add or modify collision primitives to accurately represent the shape of the model for collision detection.
  - In the Static Mesh Editor, you can assign materials and textures to the model.
  - Set up LODs (Level of Detail) if needed, to optimize performance by displaying simplified versions of the model at different distances.
- **Test and Optimize:**
  - Launch the game or simulation within Unreal Engine to test the imported model in the virtual environment.
  - Check for any issues, such as incorrect scaling, misplaced textures, or performance concerns.
  - Optimize the model if necessary by simplifying geometry, reducing texture resolution, or adjusting LOD settings to ensure smooth performance.
  - Set up LODs (Level of Detail) if needed, to optimize performance by displaying simplified versions of the model at different distances.

Below is the figure shows the model is imported from Reality Catputre to Unreal Engine.

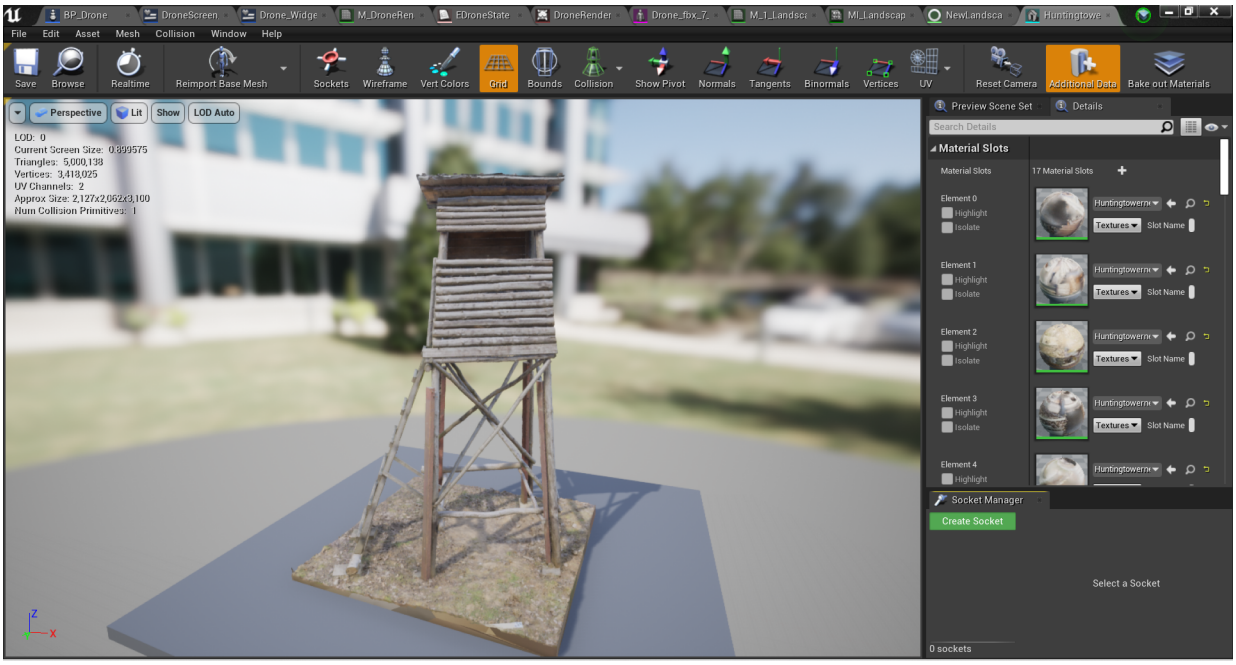


Figure 14: Imported model from Reality Capture to Unreal Engine

By following these steps, you can import models from Reality Capture into Unreal Engine, allowing you to incorporate highly detailed and realistic 3D models into your virtual environments and take advantage of the advanced features and capabilities of Unreal Engine for interactive experiences and simulations.

4.3 Unreal Engine Steps for creating drone and its navigation

In **Unreal Engine**, we created drone and navigate the drone in the same platform. Also, the Reconstruction model we obtained from the Reality Capture we imported in the Unreal Engine for the navigation of the drone. For this project we used **Unreal Engine version 4.25.4**.

Creating a drone in Unreal Engine using Blueprints involves several steps. Here's a detailed breakdown of the process:

- **Set up the Project:**
  - Launch Unreal Engine and create a new Blueprint project.
  - Choose a project template or start with a blank project.
  - Specify the project settings, including project name and directory.
- **Import Drone Assets:**

- Find or create 3D models for the drone's body, rotors, and any other components.
- Import the models into Unreal Engine by using the "Import" option in the Content Browser.
- Ensure the models are properly scaled and positioned for later use.

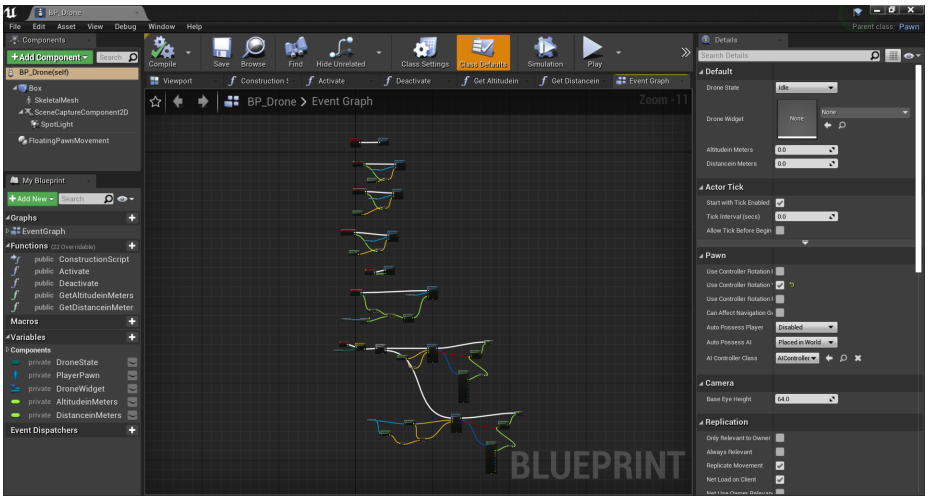


Figure 15: Drone Blueprint

- **Create the Drone Blueprint:**
  - Find or create 3D models for the drone's body, rotors, and any other components.

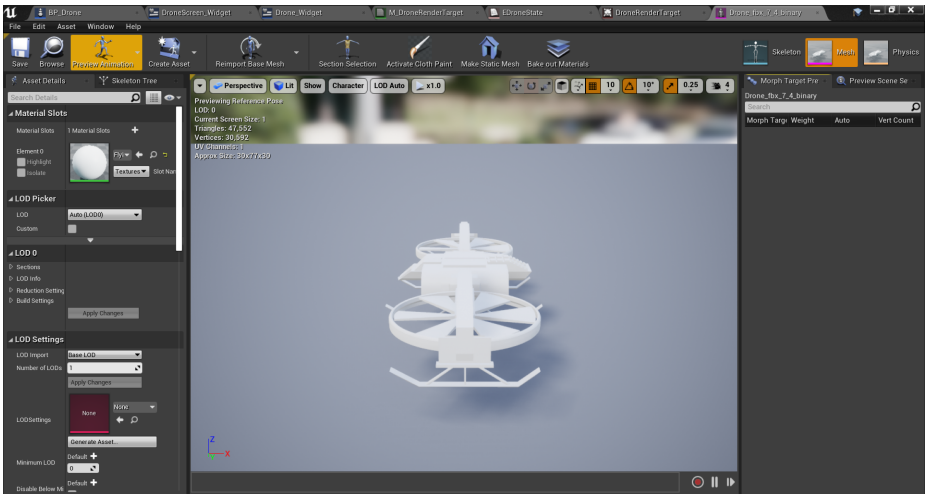


Figure 16: Drone Skeleton

– Open the Blueprint Editor by double-clicking on the Content Browser or right-clicking and selecting "Create Blueprint Class."

– Choose the desired parent class for the drone. You can start with a Pawn or Character class, depending on your requirements.

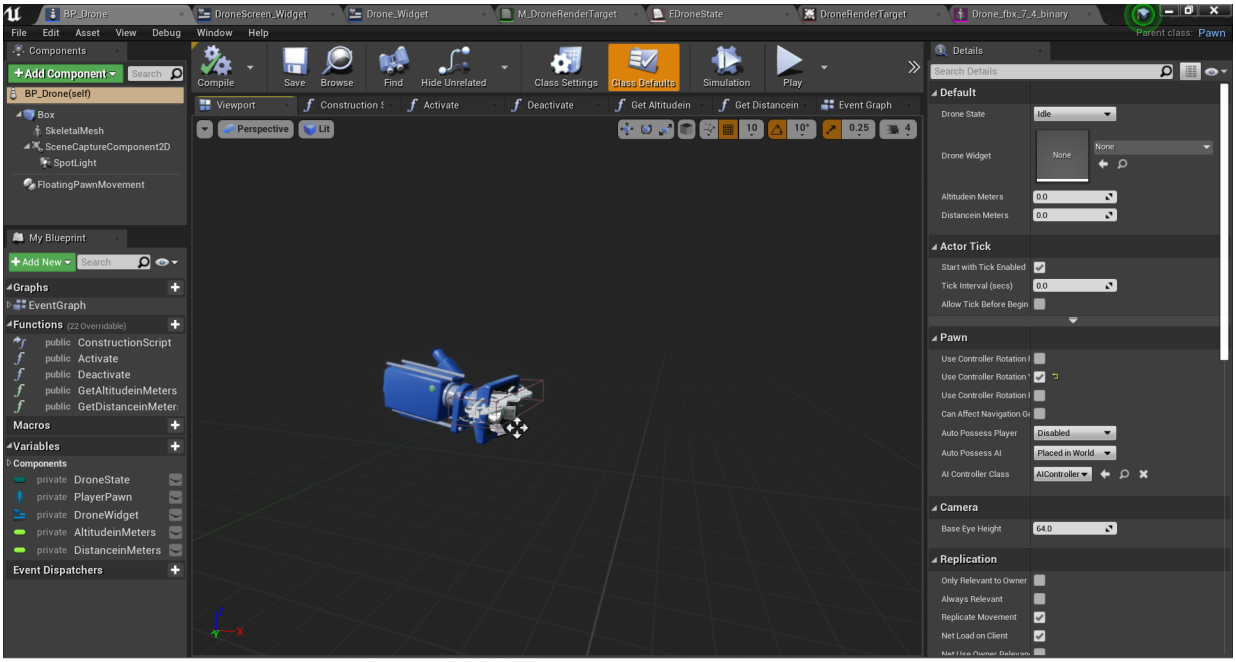


Figure 17: Camera to Drone Skeleton

– In the Blueprint Editor, you will see the Construction Script, Event Graph, and other sections.

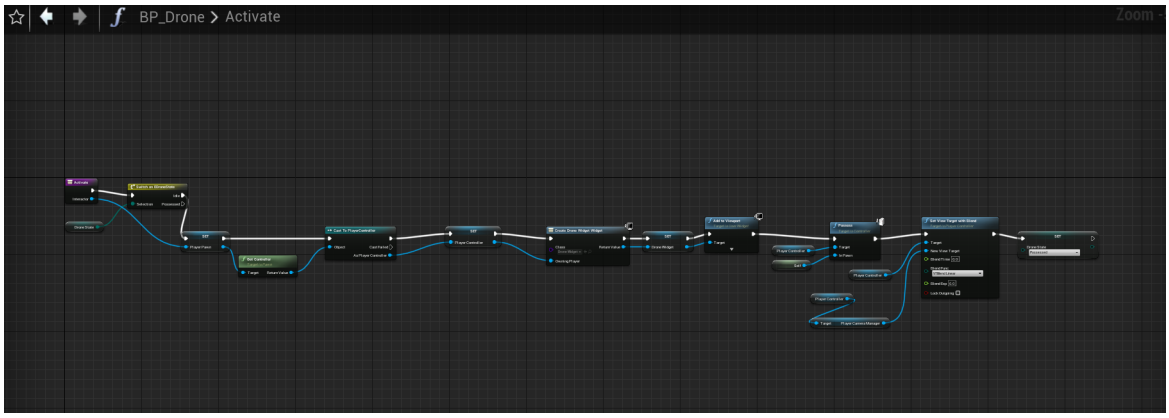


Figure 18: Drone Activation

After drone activation now we are deactivating it as shown below.

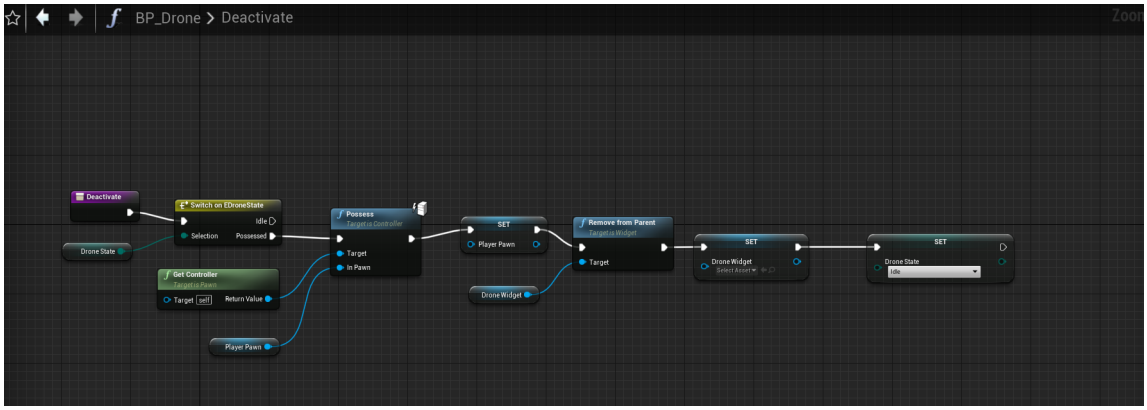


Figure 19: Drone Deactivation

Here, we attaching altitude meters to drone as shown in figure below.

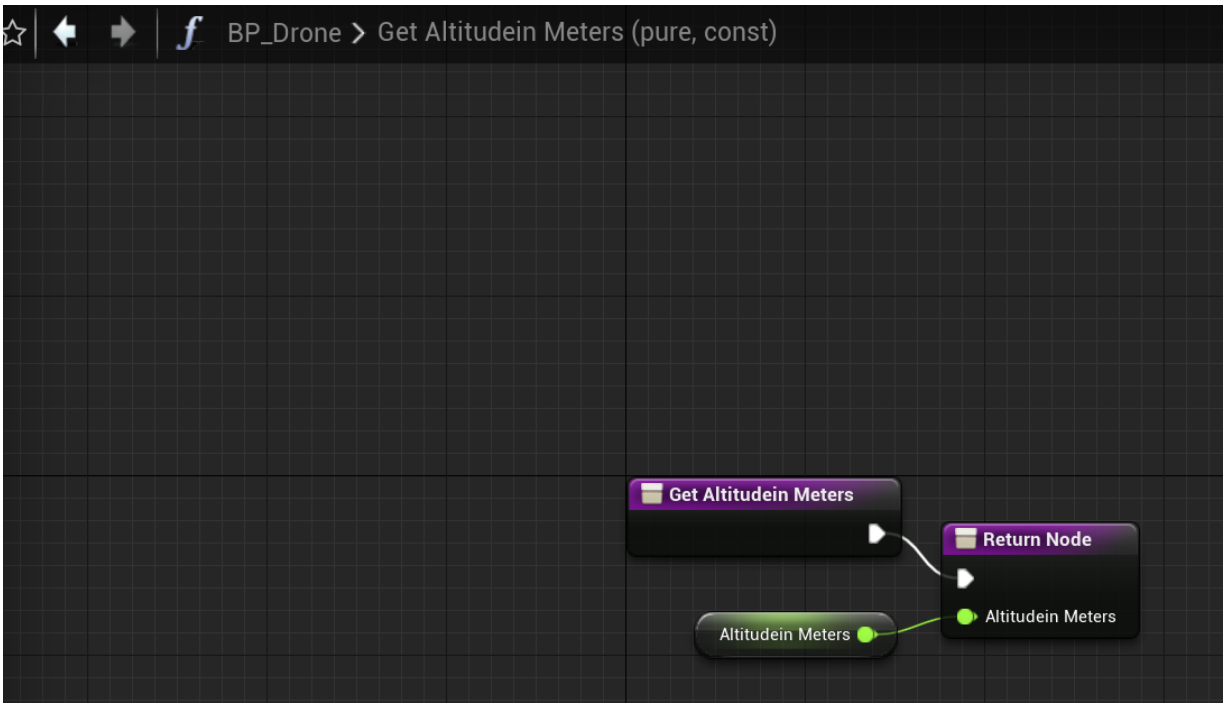


Figure 20: Drone: Get Altitude Meters (pure, const)

At the end, we get distance meters as described in figure below.

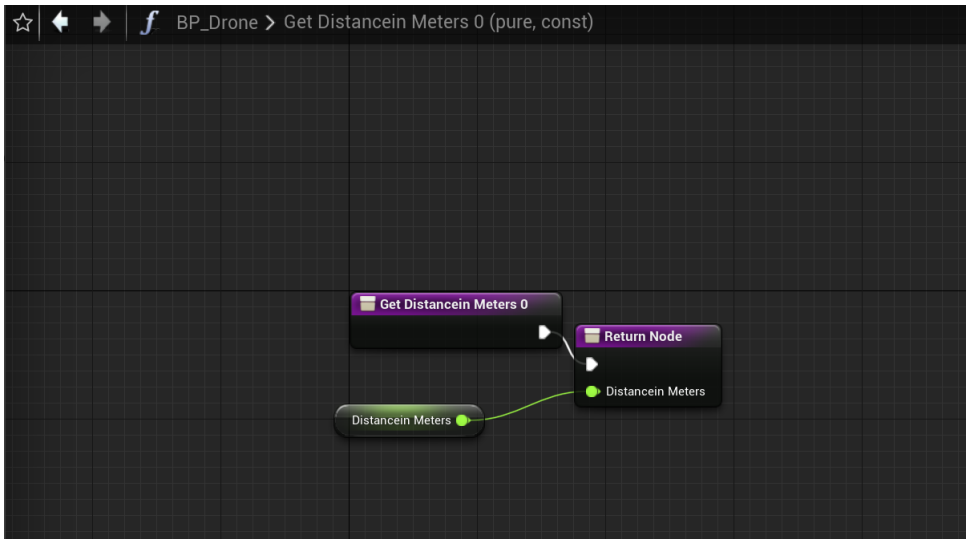


Figure 21: Drone: Get Distance Meters 0 (pure, const)

- **Design the Drone's Behavior:**
  - Use the Construction Script to set up the initial positioning and attachment of the drone components.
  - In the Event Graph, add nodes and script the desired behavior for the drone.
  - Use input events (e.g., keyboard or gamepad inputs) to control the drone's movement, such as changing its location, rotation, or velocity.
  - Implement logic for drone actions like taking off, landing, hovering, and rotating.
  - Add collision detection and response to avoid obstacles or trigger specific events.
- **Add Drone Physics:**
  - Enable physics simulation for the drone by enabling the "Simulate Physics" option in the Details panel.
  - Configure the drone's collision properties, such as collision channels, collision responses, and physical materials.
  - Adjust the drone's mass, drag, and other physical parameters to mimic realistic flight dynamics.
  - Use constraints or physics joints to connect the drone's components, such as the rotors to the body.
- **Implement Camera and View:**
  - Add a camera component to the drone Blueprint to simulate the drone's perspective as shown in figure 12.
  - Configure the camera's position, rotation, field of view, and any other desired settings.

  - Set up camera controls to allow the player to switch between different camera views or perspectives.

- **Test and Refine:**
  - Compile and save the drone Blueprint.
  - Place an instance of the drone Blueprint in the game world or level.
  - Launch the game or simulation to test the drone's behavior and controls.
  - Iterate on the design, making adjustments and refinements as needed.
  - Test the drone in different scenarios and environments to ensure its functionality and performance.

Moreover, we setup of environment blueprint as shown in figure 17.

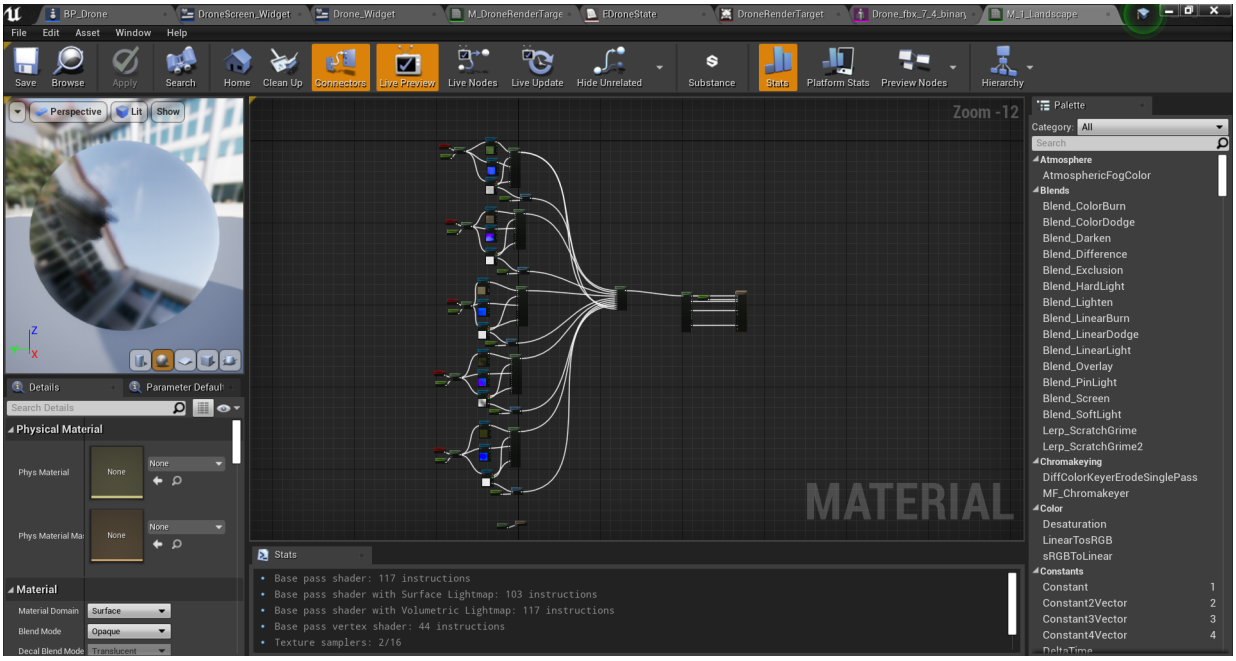


Figure 22: Environment setup Blueprint

Remember that this is a general outline of the process, and specific implementation details may vary based on your project requirements and the level of complexity you wish to achieve. The Blueprint system in Unreal Engine offers flexibility, allowing you to customize the drone's behavior and interactions according to your specific needs.

4.4 Creation of Forest Environment in Unreal Engine

To create a forest environment in Unreal Engine, you'll start by setting up the project and creating the terrain using the engine's terrain tools. Then, you'll place vegetation such as trees and bushes using foliage painting

tools. Apply realistic materials to the terrain and foliage assets to add detail and variation. Set up appropriate lighting and atmospheric effects to enhance the visual appeal. Add sound effects to create an immersive auditory experience. Figure 23 shows the forest environment in unreal engine.

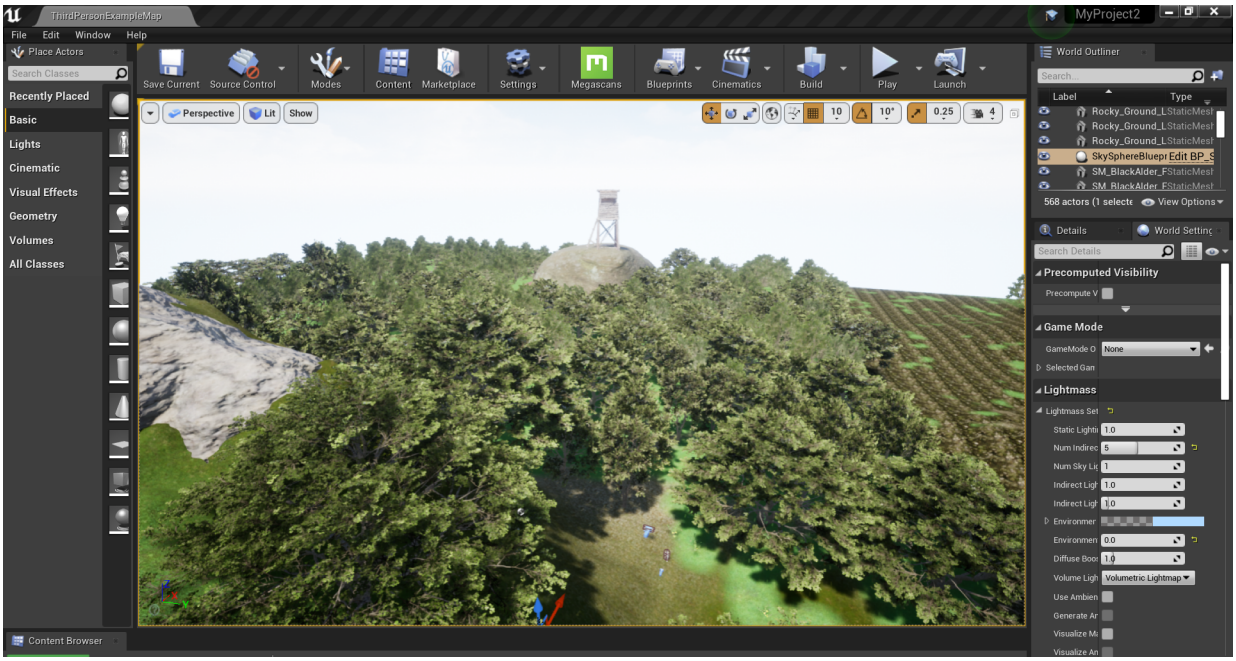


Figure 23: Forest Environment Angle1

Include additional details like rocks and streams, and incorporate interactive elements for engagement. Test and optimize the scene to ensure optimal performance. By following these steps, you can quickly create a captivating and realistic forest environment in Unreal Engine. Figure 24 shows the forest environment view from different angel in unreal engine.

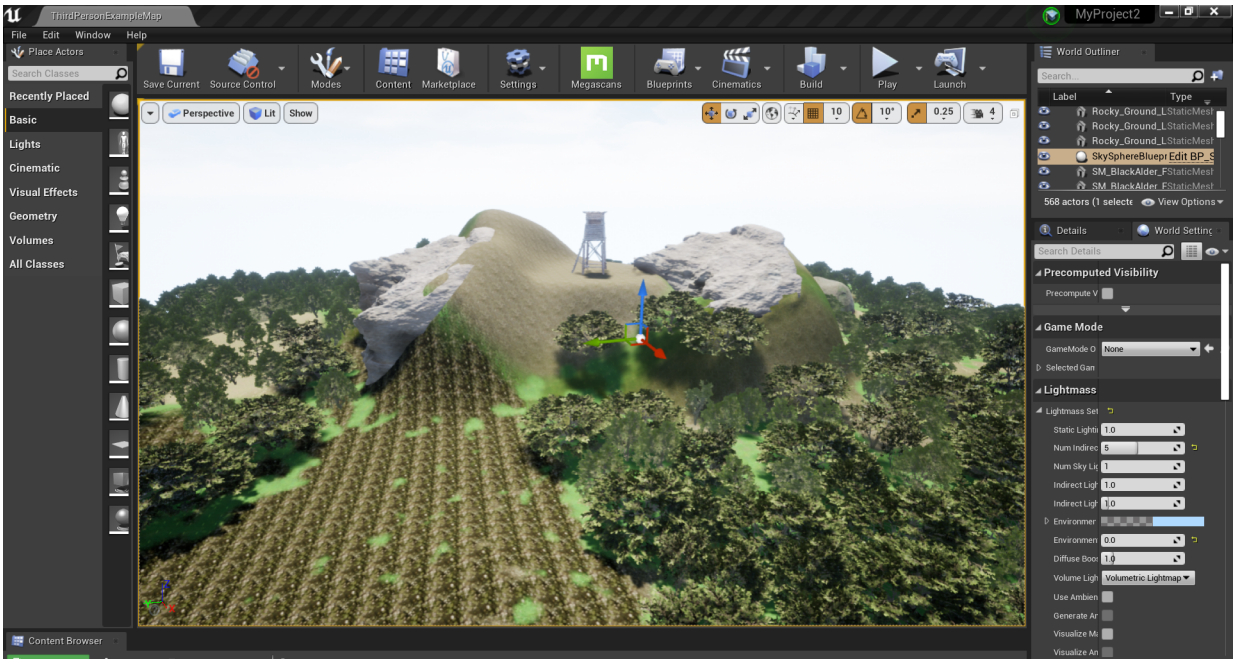


Figure 24: Forest Environment Angle2

In conclusion, creating a forest environment in Unreal Engine involves setting up the project, designing the terrain, placing vegetation, applying realistic materials, setting up lighting and atmospheric effects, adding sound effects, incorporating small details, and optimizing the scene for performance. By following these steps, you can create a visually stunning and immersive forest environment in Unreal Engine that transports users into a realistic and captivating virtual forest setting.

5 Results & Discussion

The result of the project on Photogrammetry and 3D Model Reconstruction in terms of Virtual Reality for Robotics is a highly immersive and realistic virtual environment that integrates photogrammetry-based 3D models with virtual reality technology. This integration brings several benefits and outcomes to the field of robotics:

1. **Accurate and Detailed Virtual Representations:** The project enables the creation of precise and detailed 3D models of real-world objects and scenes using photogrammetry techniques. These models capture intricate details and provide an accurate representation of the physical environment. When integrated into the virtual reality environment, they create a highly immersive experience for the users.

2. **Realistic Training and Simulation:** The resulting virtual reality environment allows for realistic training and simulation of robotic systems. Users can interact with virtual representations of robots and perform various tasks, such as manipulating objects, navigating through the virtual space, or testing different scenarios. This provides a safe and cost-effective way to train robot operators, test algorithms, and evaluate system performance.

3. **Enhanced Visualization and Understanding:** The combination of photogrammetry-based 3D models and virtual reality technology offers improved visualization and understanding of the physical environment for robotics applications. Users can explore and analyze the virtual representation of the environment, gaining insights into the spatial layout, object relationships, and potential obstacles. This aids in planning and decision-making processes for robotics tasks.

4. **Seamless Human-Robot Interaction:** The project's outcome facilitates seamless human-robot interaction within the virtual reality environment. Users can intuitively control and communicate with virtual robots, providing commands or performing actions through natural gestures or user interfaces. This enables the development and testing of human-robot interaction algorithms and enhances the overall user experience.

5. **Robust System Development and Evaluation:** The virtual reality environment resulting from the project serves as a valuable platform for the development and evaluation of robotic systems. Researchers and developers can test and refine algorithms, control strategies, and sensor configurations within the virtual environment, prior to real-world implementation. This helps in identifying and addressing potential issues or limitations before deployment.

6. **Collaborative Robotics Research:** The virtual reality environment allows for collaborative research and development in the field of robotics. Multiple users can connect to the same virtual space simultaneously, enabling collaboration, knowledge sharing, and joint experimentation. This fosters interdisciplinary collaboration and accelerates innovation in robotics.

Figure below shows the **content browser** folder which is obtained at end of our project.

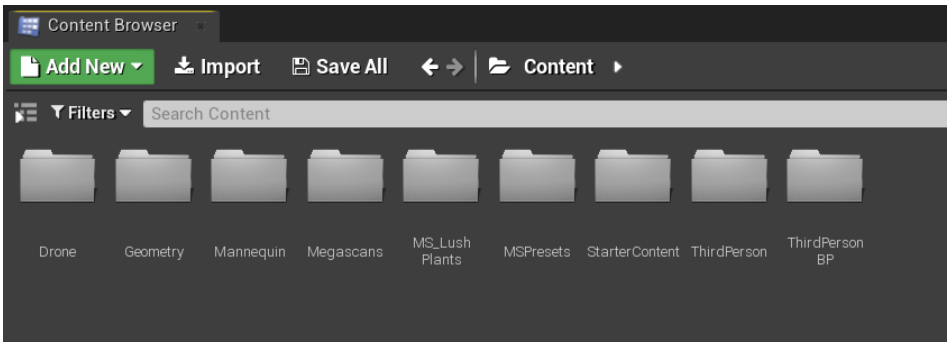


Figure 25: Content Browser Folder

Figure below depicts when the game starts in Unreal Engine.

Figure 26: Game Start

Figure below highlights with the drone's camera view which shows our model which we imported from reality Capture.

Figure 27: Drone: Camera View of 3D Model from Reality Capture

Figure below explains about the drone camera which shows the environment.

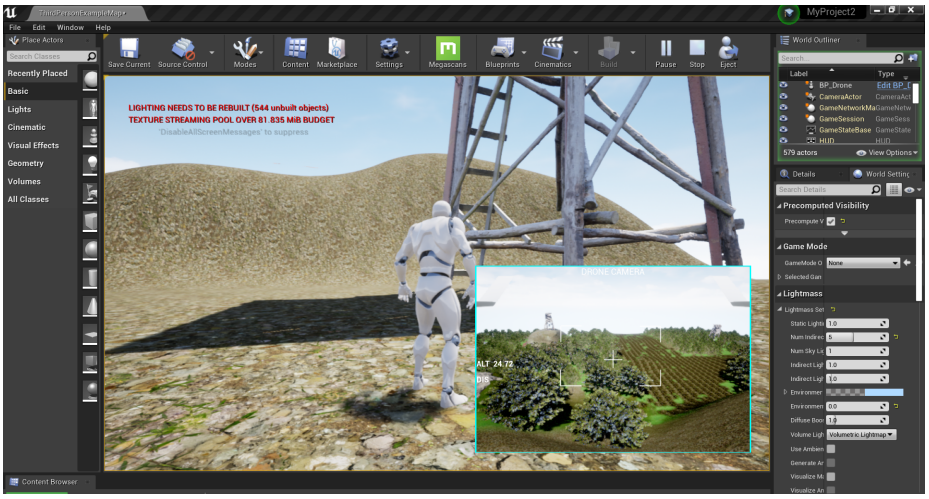


Figure 28: Drone Camera: Environment View

Overall, the project's result combines the power of photogrammetry and 3D model reconstruction with virtual reality technology to enhance robotics applications. The outcome is a realistic and immersive virtual environment that facilitates training, simulation, visualization, and interaction with robotic systems. It opens up new possibilities for research, development, and deployment of robotics technologies in various domains, including manufacturing, healthcare, exploration, and more.

6 Conclusions

In conclusion, the project on Photogrammetry and 3D Model Reconstruction in terms of Virtual Reality for Robotics has demonstrated the potential of combining these technologies to advance the field of robotics. By leveraging photogrammetry techniques to create accurate 3D models and integrating them into a virtual reality environment, several significant outcomes have been achieved.

The project has enabled the creation of highly immersive and realistic virtual environments, where robotics applications can be simulated, tested, and trained. The accurate and detailed 3D models reconstructed through photogrammetry techniques have provided a precise representation of the physical environment, enhancing visualization and understanding.

The resulting virtual reality environment has proven to be a valuable platform for realistic training and simulation of robotic systems. It has offered a safe and cost-effective way to train robot operators, test algorithms, and evaluate system performance, reducing the need for physical prototypes and potential risks associated with real-world testing.

Furthermore, the seamless human-robot interaction within the virtual environment has opened up opportunities for intuitive control and communication with virtual robots. This has facilitated the development and testing of human-robot interaction algorithms and enhanced the overall user experience.

The project's outcomes have also supported robust system development and evaluation, providing researchers and developers with a platform to refine algorithms, control strategies, and sensor configurations before real-world implementation. It has helped identify potential issues and limitations, leading to more reliable and efficient robotic systems.

Additionally, the virtual reality environment has encouraged collaborative research and development, allowing multiple users to connect and collaborate within the same virtual space. This collaborative aspect has accelerated innovation in robotics and fostered interdisciplinary collaboration.

Overall, the project's conclusion highlights the potential of combining photogrammetry and 3D model reconstruction with virtual reality for robotics applications. The achieved outcomes pave the way for advancements in training, simulation, visualization, and human-robot interaction, leading to more sophisticated and efficient robotics systems across various industries.

References

[1] RealityCapture - Photogrammetry Software — capturingreality.com. https://www.capturingreality.com/realitycapture.

[2] RealityCapture - Tutorial. https://www.youtube.com/watch?v=WrCOhes1Zgg.

[3] Unreal Engine - Tutorial. https://www.youtube.com/watch?v=TASHWpdQFwc&list=RDCMUCwt9gg3t19wDNY1TVpm97sw&index=2.

[4] Unreal Engine | The most powerful real-time 3D creation tool — unrealengine.com. https://www.unrealengine.com/en-US.

[5] Ph.D. Florent Poux. The Ultimate Guide to 3D Reconstruction with Photogrammetry — towardsdatascience.com. https://towardsdatascience.com/the-ultimate-guide-to-3d-reconstruction-with-photogrammetry-56155516ddc4. [Accessed 08-Jan-2023].

[6] David Novotny Georgia Gkioxari, Shubham Tulsiani. Pushing state-of-the-art in 3D content understanding — ai.facebook.com. https://ai.facebook.com/blog/pushing-state-of-the-art-in-3d-content-understanding/, February 18, 2019.

[7] Georgios Kordelas, Juan Agapito, Jesús Vegas, and Petros Daras. State-of-the-art algorithms for complete 3d model reconstruction. 09 2010.

[8] Malgorzata Kujawinska, Robert Sitnik, Michal Pawlowski, Piotr Garbat, and Marek Wegiel. Three-dimensional data acquisition and processing for virtual reality applications. *Proceedings of SPIE - The International Society for Optical Engineering*, 4778, 06 2002.

[9] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. State of the art on 3d reconstruction with rgb-d cameras. *Computer Graphics Forum*, 37(2):625–652, 2018.